Roteamento distribuído baseado em políticas sem coordenação global

SBRC 2019

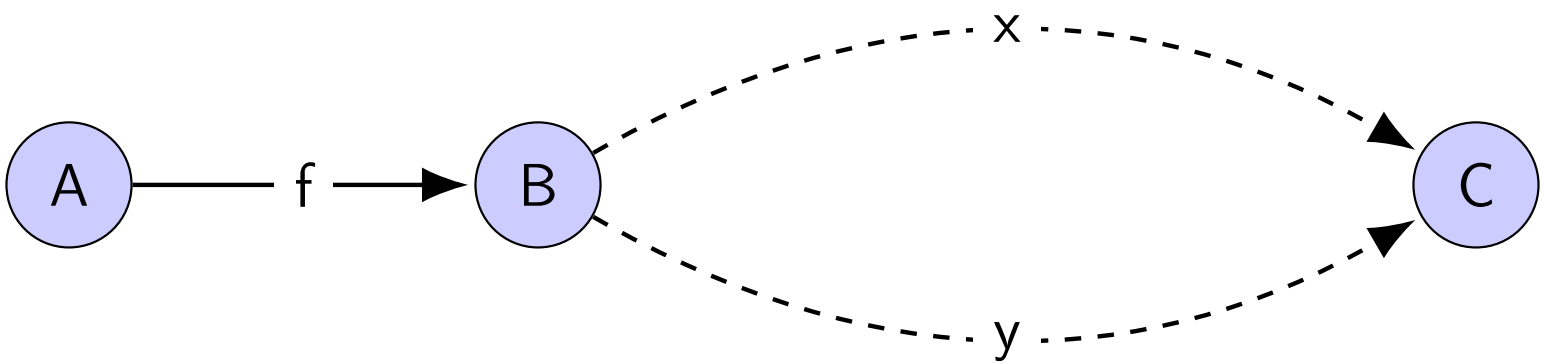
Timothy G. Griffin
tgg22@cam.ac.uk

University of Cambridge

May 9, 2019

Route selection

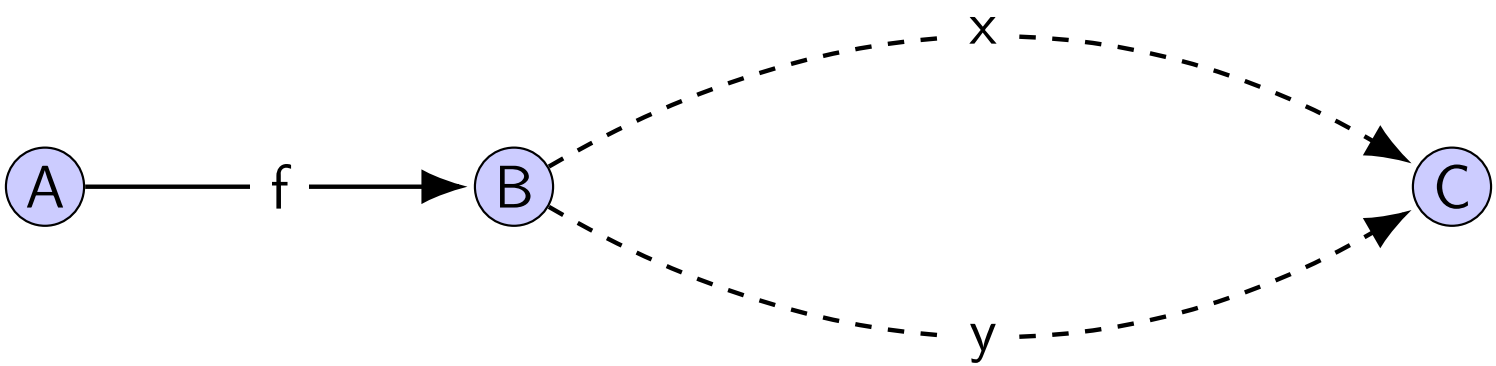
- node B has two routes x and y to node C
- $\oplus$ selects the best route
- selection at node B is $x \oplus y$
- selection at node A?
  - $f(x \oplus y)$ or $f(x) \oplus f(y)$?



Distributivity

The mathematics behind most routing algorithms assumes distributivity.

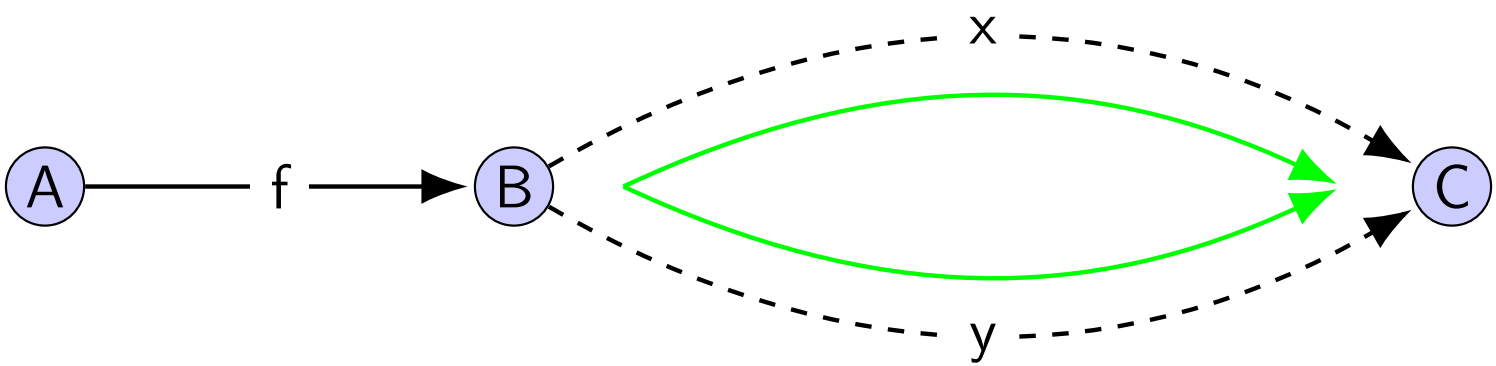
$$f(x \oplus y) = f(x) \oplus f(y)$$



Distributivity

The mathematics behind most routing algorithms assumes distributivity.

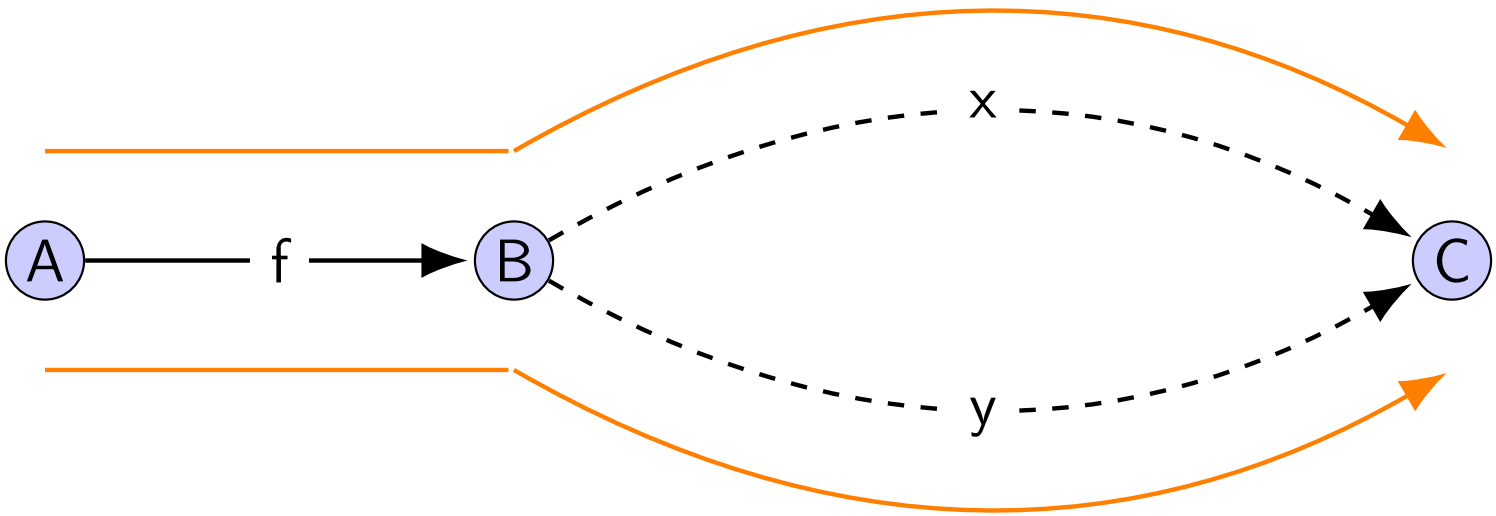
$$f(x \oplus y) = f(x) \oplus f(y)$$



Distributivity

The mathematics behind most routing algorithms assumes distributivity.

$$f(x \oplus y) = f(x) \oplus f(y)$$



Distributivity

A policy language is distributive iff for every policy f and routes x and y

$$f(x \oplus y) = f(x) \oplus f(y)$$

Advantages:

- Global optimum!

Disadvantages:

- Very restrictive as to the problems you can solve.

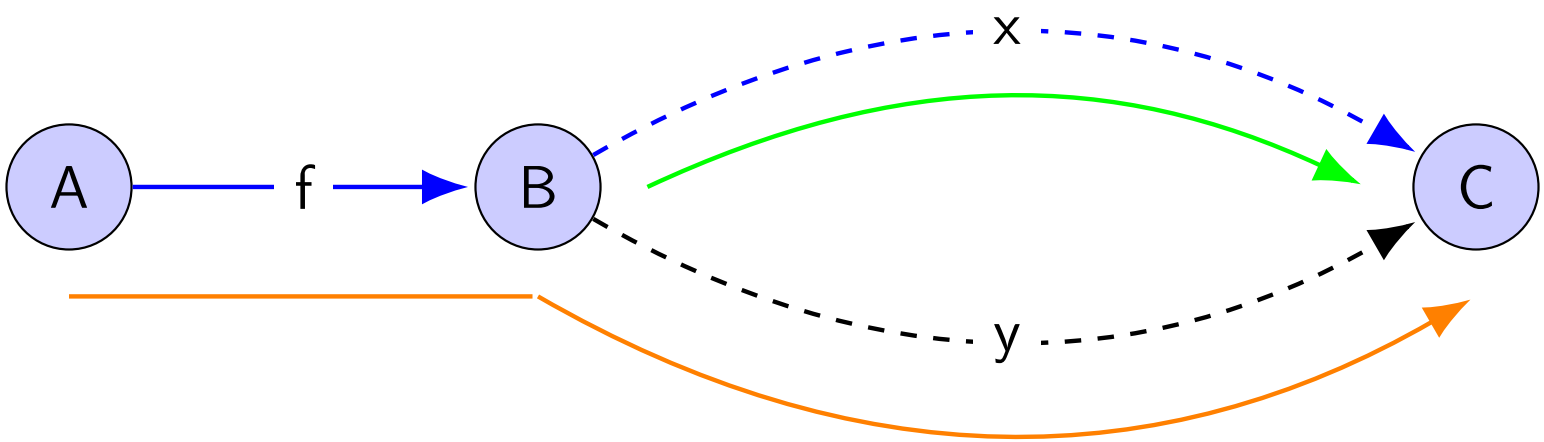
policy-rich

We will define policy-rich language as one that allows us to violate distributivity.

Example of violating distributivity

Shortest-paths with "no more than one blue edge".

- Path x has one blue edge in it
- Path y has no blue edges in it
- The edge A to B is blue
- Node B prefers route x, i.e. $x \oplus y = x$



Conditionals : An easy way of violating distributivity

Suppose g and h are policy functions and P is a predicate on routes (such as "true if route has one blue edge").
Suppose f is defined as

$$f(r) = \text{if } P(r) \text{ then } g(r) \text{ else } h(r)$$

Suppose that:

$$\begin{array}{lcl} P(x) & = & \text{true} \\ P(y) & = & \text{false} \\ x \oplus y & = & x \\ g(x) \oplus h(y) & = & h(y) \end{array}$$

Then

$$f(x \oplus y) = f(x) = g(x)$$

and

$$f(x) \oplus f(y) = g(x) \oplus h(y) = h(y)$$

Related work: Gao & Rexford 2001

Stable Internet Routing Without Global Coordination

Lixin Gao, *Member, IEEE*, and Jennifer Rexford, *Senior Member, IEEE*

***Abstract*—The Border Gateway Protocol (BGP) allows an autonomous system (AS) to apply diverse local policies for selecting routes and propagating reachability information to other domains. However, BGP permits ASs to have conflicting policies that can lead to routing instability. This paper proposes a set of guidelines for an AS to follow in setting its routing policies, without requiring coordination with other ASs. Our approach exploits the Internet's hierarchical structure and the commercial relationships between ASs to impose a partial order on the set of routes to each destination. The guidelines conform to conventional traffic-engineering practices of ISPs, and provide each AS with significant flexibility in selecting its local policies. Furthermore, the guidelines ensure route convergence even under changes in the topology and routing policies. Drawing on a formal model of BGP, we prove that following our proposed policy guidelines guarantees route convergence. We also describe how our methodology can be applied to new types of relationships between ASs, how to verify the hierarchical AS relationships, and how to realize our policy guidelines. Our approach has significant practical value since it preserves the ability of each AS to apply complex local policies without divulging its BGP configurations to others.**

***Index Terms*—Border Gateway Protocol (BGP), convergence, Internet, protocols, routing.**

I. Introduction

THE INTERNET connects thousands of Autonomous Systems (ASs) operated by different institutions, such as Internet Service Providers (ISPs), companies, and universities. Routing within an AS is controlled by intradomain protocols such as OSPF, IS–IS, and RIP [1]. ASs interconnect via dedicated links and public network access points, and exchange reachability information using the Border Gateway Protocol (BGP) [2], [3]. BGP is an interdomain routing protocol that allows ASs to apply local policies for selecting routes and propagating routing information, without revealing their policies or internal topology to others. However, recent studies have shown that a collection of ASs may have conflicting BGP policies that lead to route divergence [4], [5]. Route divergence can result in route oscillation, which can significantly degrade the end-to-end performance of the Internet. Avoiding these conflicting BGP policies is crucial for the stability of the Internet routing infrastructure. Yet, to be practical, any technique for ensuring convergence should not sacrifice the ability of each AS to apply complex local policies.

A natural approach to the route convergence problem involves the use of the Internet Routing Registry, a repository of routing policies specified in a standard language [6]. A complete and up-to-date registry could check if the set of routing policies has any potential convergence problems. However, this global coordination effort faces several impediments. First, many ISPs may be unwilling to reveal their local policies to others, and may not keep the registry up-to-date. Second, and perhaps more importantly, even if ISPs decide to reveal their local polices, recent work has shown that statically checking for convergence properties is an NP-complete problem [4]. Third, even if the registry could ensure convergent routes under a given topology, BGP still might not converge under router or link failures, or a policy change. Hence, rather than requiring global coordination, we believe that convergence should be achieved by restricting the set of policies that each AS can apply. In this paper, we propose a set of guidelines for an AS to follow in setting its routing policies, without requiring coordination with other ASs [7]. In addition, the guidelines ensure routing convergence even under changes in the underlying topology (e.g., router or link failure) or the routing policies.

Our approach capitalizes on the Internet's hierarchical structure and the commercial relationships between ASs. These relationships include customer–provider, peer-to-peer, and backup [8], [9]. A customer pays its provider for connectivity to the rest of the Internet, whereas peers agree to exchange traffic between their respective customers free of charge; an AS may also provide backup connectivity to the Internet in the event of a failure. To ensure route convergence, we impose a partial order on the set of routes to each destination. Under our guidelines, routing via a peer or a provider is never preferable to routing via a customer link; furthermore, routes via backup links have the lowest preference. An AS is free to apply any local policies to the routes learned from neighbors within each preference class. These guidelines conform to conventional traffic-engineering practices of ISPs, and this might well explain why Internet routing divergence has not occurred yet. However, it is crucial to make these guidelines explicit since BGP itself does not constrain routing policies to ensure convergence. Based on our results, we propose a simple routing registry that stores only the *relationship* between each AS pair, rather than the entire set of routing policies. These relationships can be explicitly registered by the ASs or inferred from the BGP routing tables available throughout the Internet [10], [11].

The remainder of the paper is structured as follows. Section II presents an overview of interdomain routing and discusses previous work on BGP protocol dynamics. Then, Section III presents a formal model of BGP that includes ASs with

Manuscript received September 14, 2000; revised July 1, 2001; recommended by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Calvert. The work of L. Gao was supported in part by the National Science Foundation under Grants ANI-9977555 and ANI-0085848, and NSF CAREER Award Grant ANI-9875513.

L. Gao is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA (e-mail: lgao@ecs.umass.edu).

J. Rexford is with the Internet and Networking Systems Center, AT&T Labs — Research, Florham Park, NJ 07932 USA (e-mail: jrex@research.att.com).

Publisher Item Identifier S 1063-6692(01)10544-3.

Gao & Rexford 2019

Related work: João L. Sobrinho 2005

An Algebraic Theory of Dynamic Network Routing

João Luís Sobrinho, *Member, IEEE*

***Abstract*—We develop a non-classic algebraic theory for the purpose of investigating the convergence properties of dynamic routing protocols. The algebraic theory can be regarded as a generalization of shortest-path routing, where the new concept of free cycle generalizes that of a positive-length cycle. A primary result then states that routing protocols always converge, though not necessarily onto optimal paths, in networks where all cycles are free. Monotonicity and isotonicity are two algebraic properties that strengthen convergence results. Monotonicity implies protocol convergence in every network, and isotonicity assures convergence onto optimal paths.**

A great many applications arise as particular instances of the algebraic theory. In intra-domain routing, we show that routing protocols can be made to converge to shortest and widest paths, for example, but that the composite metric of Internet Gateway Routing Protocol (IGRP) does not lead to optimal paths. The more interesting applications, however, relate to inter-domain routing and its Border Gateway Protocol (BGP), where the algebraic framework provides a mathematical template for the specification, design, and verification of routing policies. We formulate existing guidelines for inter-domain routing in algebraic terms, propose new guidelines contemplating backup relationships between domains, and derive a sufficient condition for signaling correctness of internal-BGP.

***Index Terms*—Algebra, convergence, inter-domain routing, intra-domain routing, routing protocols.**

I. Introduction

NON-CLASSIC algebra has made headway in many branches of electrical engineering and computer science, from coding and cryptography to compiler design and networking, unifying seemingly unrelated concepts and establishing fundamental results. Can it also shed light into distributed network routing, especially as witnessed in the Internet protocols? We answer affirmatively by defining suitable algebraic structures and exploring their properties.

Packets in the Internet are forwarded by routers as a function of their destinations, regardless of their origins. The forwarding table at each router is kept up to date by dynamic routing protocols that react to network failures and additions. Routing is administratively divided in intra-domain and inter-domain, each with its own set of goals and protocols. A domain is a collection of routers under the administrative and operational control of a single entity. Intra-domain routing is, in general, performance-oriented: the entity administering the domain aims at the best use of its internal network resources. Routing Information Protocol (RIP) [1], [2], Interior Gateway Routing Protocol (IGRP) [3], and Open Shortest Path First (OSPF) [4], [5] are examples of intra-domain routing protocols, the first and second of distance-vector type, and the third of link-state type [6]. By contrast, inter-domain routing is policy-oriented: the various domains forward packets toward the destinations as a function of local policies that reflect the commercial relationships established between them. The Border Gateway Protocol (BGP) [7], [8] is currently the only protocol for inter-domain routing and is a path-vector protocol [6].

The purpose of the algebraic theory herein presented is to establish fundamental results on the convergence of routing protocols, and see them applied in a variety of different environments. The convergence results are formulated in terms of path-vector protocols, but they bear as well to distance-vector protocols. Link-state protocols call for a more specific, less general, algebraic theory, which is presented in [9].

An algebra for routing comprises a set of labels, a set of signatures, and a set of weights. Each network link has a label and each network path has a signature. There is an operation to obtain the signature of a path from the labels of its constituent links, and a function mapping signatures to weights. Ultimately, each path will have a weight, and these weights are ordered such that any set of paths with the same origin and destination can be compared: the lower the weight of a path the more preferred the path is. For example, if labels, signatures, and weights are real numbers, composed by standard addition and ordered by the standard less-than-or-equal relation, the resulting instance of the algebra represents shortest-path routing. In light of this, the algebraic theory can be seen as providing a generalization of shortest-path routing.

A central concept is that of free cycle. In a free cycle, for any collection of paths, each starting at a different node of the cycle, at least one of these paths weighs less than the path that starts at the same node, proceeds to the node's neighbor around the cycle, and continues with the path that starts at the neighbor. Intuitively, if a cycle is free, then, given any destination in the network, at least one of its nodes forwards packets to the destination out of the cycle, instead of around the cycle, thus preventing packets from being trapped in a loop. In the instance of the algebra representing shortest-path routing, the free cycles are exactly the positive-length cycles, so that the former cycles generalize the latter. Moreover, the generalization carries over to the role those cycles play on the convergence of routing protocols. In shortest-path routing, it is known that path-vector protocols converge if all network cycles have positive length [10]. For the broader algebraic theory, we show that path-vector protocols converge if all network cycles are free.

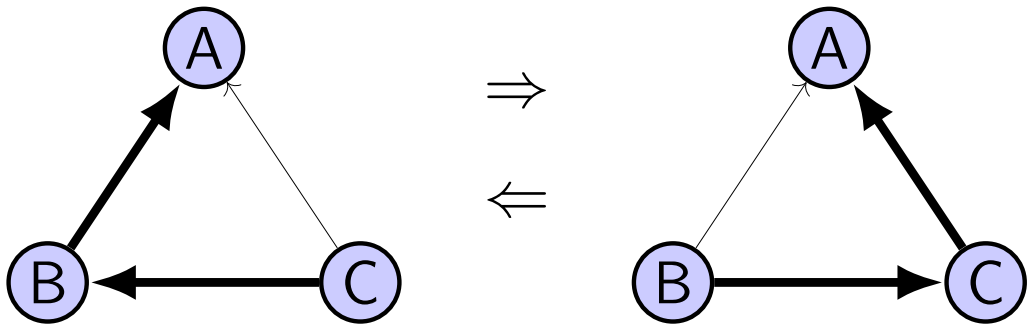
Some algebras are enriched by properties that intertwine its elements, allowing for stronger statements about protocol

Manuscript received February 16, 2004; revised October 8, 2004, and October 19, 2004; approved by IEEE/ACM Transactions on Networking Editor J. Crowcroft.

The author is with the Instituto de Telecomunicações, Instituto Superior Téc-

Problems we want to avoid

- Routing oscillations resulting in non-convergence



Strictly increasing

An algebra is *strictly increasing* if extending a route makes it worse:

$$x < f(x)$$

where $x \leq y \triangleq x \oplus y = x$, $\quad x < y \triangleq x \leq y$ and $x \neq y$.

Sobrinho showed path-vectoring (distributed Bellman-Ford where paths are used to avoid loops) always converge.

Conditional policies ("route maps") preserve this property!
If g and h are both strictly increasing then

$$f(r) = \text{if } P(r) \text{ then } g(r) \text{ else } h(r)$$

is also strictly increasing.

Today's talk : A summary of some recent results

[ICNP 2018] Rate of convergence of increasing path-vector routing protocols.
Matthew L. Daggitt and Timothy G. Griffin.
International Conference on Network Protocols (ICNP) 2018

[SIGCOMM 2018] Asynchronous Convergence of Policy-Rich Distributed Bellman-Ford Routing Protocols.
Matthew L. Daggitt, Alexander J. T. Gurney, Timothy Griffin.
SIGCOMM 2018.

[ITP 2018] An Agda Formalization of Üresin & Dubois' Asynchronous Fixed-Point Theory.
Ran Zmigrod, Matthew L. Daggitt and Timothy G. Griffin.
Interactive Theorem Proving 2018.

Our Main Contributions

For *strictly-increasing* path-vector protocols:

- The first polynomial bound on convergence.
- Proof that protocols converge to a **unique** solution
- A new proof technique that exploits the 1990 results of Üresin & Dubois' asynchronous fixed-point theory
- All proofs are entirely formalised in the theorem prover Agda

A small Agda fragment

```
p[FXᵢⱼ]≈[]⟹i≡j : ∀ X i j → path (F X i j) ≈ₚ valid [] → i ≡ j
p[FXᵢⱼ]≈[]⟹i≡j X i j p[FXᵢⱼ]≈[] with FXᵢⱼ≈Aᵢₖ▷Xₖⱼ⊎Iᵢⱼ X i j
... | inj₂ FXᵢⱼ≈Iᵢⱼ          = p[Iᵢⱼ]≈[]⟹i≡j (≈ₚ-trans (path-cong (≈-sym FXᵢⱼ≈Iᵢⱼ)) p[FXᵢⱼ]≈[])
... | inj₁ (k , FXᵢⱼ≈AᵢₖXₖⱼ) with A i k ▷ X k j ≟ ∞#
...   | yes AᵢₖXₖⱼ≈∞ = contradiction
  (≈ₚ-trans (≈ₚ-trans (≈ₚ-sym (r≈∞⟹path[r]≈∅ AᵢₖXₖⱼ≈∞)) (path-cong (≈-sym FXᵢⱼ≈AᵢₖXₖⱼ))) p[FXᵢⱼ]≈[]) λ()
...   | no  AᵢₖXₖⱼ≉∞ with path (X k j) | inspect path (X k j)
...       | invalid | [ p[Xₖⱼ]≡∅ ] = contradiction (p[r]≡∅⟹f▷r≈∞ (A i k) p[Xₖⱼ]≡∅) AᵢₖXₖⱼ≉∞
...       | valid q | [ p[Xₖⱼ]≡q ] with ≈ₚ-reflexive p[Xₖⱼ]≡q | (i , k) ↔ᵥ? q | i ∉ᵥₚ? q
...         | pᵣ≈q | no ¬ik↔q | _       = contradiction (path-reject (A i k) pᵣ≈q (inj₁ ¬ik↔q)) AᵢₖXₖⱼ≉∞
...         | pᵣ≈q | _       | no  i∈q = contradiction (path-reject (A i k) pᵣ≈q (inj₂ i∈q))   AᵢₖXₖⱼ≉∞
...         | pᵣ≈q | yes ik↔q | yes i∉q = contradiction (begin
  valid (_ ∷ q ∣ _ ∣ _) ≈⟨ ≈ₚ-sym (path-accept (A i k) pᵣ≈q AᵢₖXₖⱼ≉∞ ik↔q i∉q) ⟩
  path (A i k ▷ X k j)  ≈⟨ ≈ₚ-sym (path-cong FXᵢⱼ≈AᵢₖXₖⱼ) ⟩
  path (F X i j)        ≈⟨ p[FXᵢⱼ]≈[] ⟩
  valid []              ∎) λ {(valid ())}
  where open EqReasoning (𝕡ₛ n)
```

Vision, Hope, (Wishful Thinking?) ...

- BGP is a legacy system, probably no hope there!
- We hope our results will be useful in future routing protocol design.
- These results my find applications outside of networking...

Routing Algebra

A *routing algebra*[1] is a tuple $(S,\ \oplus,\ F,\ \overline{0},\ \overline{\infty})$ where:

- S is the set of weights
- $\oplus : S \times S \to S$ is the choice operator
- $F \subseteq S \to S$ is the set of edge functions
- $\overline{0} \in S$ is the weight of the trivial route
- $\overline{\infty} \in S$ is the weight of the invalid route

[1] Inspired by Sobrinho 2005

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

The following properties are assumed

Property	Definition
Operator $\oplus$ is a choice	$a \oplus b \in \{a, b\}$
Order of choices doesn't matter	$a \oplus (b \oplus c) = (a \oplus b) \oplus c$
Order of routes doesn't matter	$a \oplus b = b \oplus a$
The trivial route is the best route	$a \oplus \overline{0} = \overline{0}$
The invalid route is the worst route	$a \oplus \overline{\infty} = a$
Extending the invalid route is invalid	$f(\overline{\infty}) = \overline{\infty}$

What problem are we solving?

Find matrix $\mathbf{X}$ so that

$$\mathbf{X} = \mathbf{A}(\mathbf{X}) \oplus \mathbf{I}$$

where

$$\mathbf{A}_{ij} \triangleq i\text{'s policy for neighbour } j$$

$$(\mathbf{X} \oplus \mathbf{Y})_{ij} \triangleq \mathbf{X}_{ij} \oplus \mathbf{Y}_{ij}$$

$$\mathbf{I}_{ij} \triangleq \begin{cases} \overline{0} & \text{if } i = j \\ \overline{\infty} & \text{otherwise} \end{cases}$$

$$\mathbf{A}(\mathbf{X})_{ij} \triangleq \bigoplus_k \mathbf{A}_{ik}(\mathbf{X}_{kj})$$

How to find a solution?

$$\sigma(\mathbf{X}) \triangleq \mathbf{A}(\mathbf{X}) \oplus \mathbf{I}$$

Iterate σ

$$\begin{aligned} \sigma^0(\mathbf{X}) &\triangleq \mathbf{X} \\ \sigma^{k+1}(\mathbf{X}) &\triangleq \sigma(\sigma^k(\mathbf{X})) \end{aligned}$$

This is a very abstract and synchronous model of how distributed Bellman-Ford computes solutions.

How to find a solution?

A state $\mathbf{X}$ is *stable* if it is a fixed point for σ:

$$\sigma(\mathbf{X}) = \mathbf{X}$$

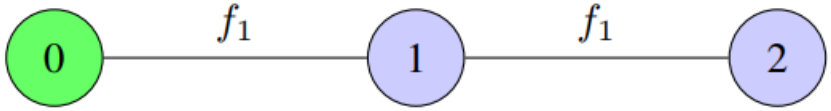
This is equivalent to saying that no node can improve its selected routes by unilaterally choosing to switch. Therefore such a state is a *local* but not necessarily a *global* optimum.

The computation σ converges synchronously from an arbitrary state $\mathbf{X}$ if there exists a t such that:

$$\sigma^{t+1}(\mathbf{X}) = \sigma^{t}(\mathbf{X})$$

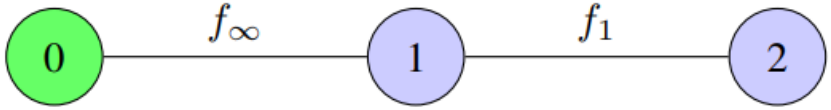
where $\sigma^{t}(\mathbf{X})$ is necessarily a stable state.

Count-to-infinity, Count-to-convergence



	Nodes		
Time	0	1	2
0	**0**	∞	∞
1	-	**1**	∞
2	-	-	**2**

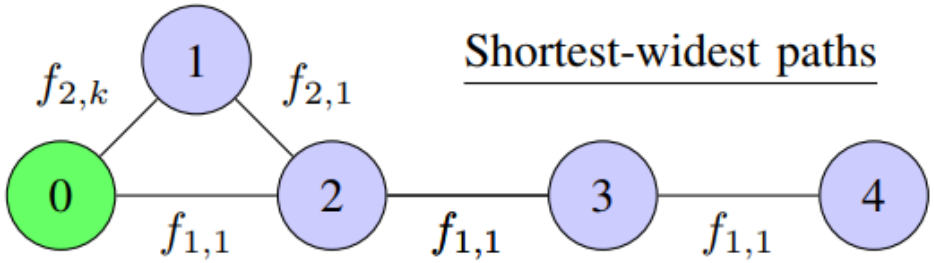
$\Downarrow$ – link $(1, 0)$ goes down



	Nodes		
Time	0	1	2
0	**0**	1	2
1	-	3	2
2	-	3	4
3	-	5	4
4	-	5	6
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Count-to-convergence: strictly increasing

In strictly increasing algebras count-to-convergence can occur even **without** changes to the network topology.



Time	0	1	2	3	4
0	$(\infty, \mathbf{0})$	$(0, \infty)$	$(0, \infty)$	$(0, \infty)$	$(0, \infty)$
1	-	$(\mathbf{2}, \mathbf{k})$	$(1, 1)$	$(0, \infty)$	$(0, \infty)$
2	-	-	$(\mathbf{2}, \mathbf{k+1})$	$(1, 2)$	$(0, \infty)$
3	-	-	-	$(1, k+2)$	$(1, 3)$
4	-	-	-	$(1, 4)$	$(1, k+3)$
5	-	-	-	$(1, k+2)$	$(1, 5)$
6	-	-	-	$(1, 6)$	$(1, k+3)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	-	-	-	$(1, k)$	$(1, k+3)$
$k+1$	-	-	-	$(\mathbf{1}, \mathbf{k+2})$	$(1, k+1)$
$k+2$	-	-	-	$(1, k+2)$	$(\mathbf{1}, \mathbf{k+3})$

Path-vector protocols

Path vector protocols fix count-to-convergence.

- Path weights track the path along which they were generated.
- Path weights with loops in are discarded.
- In distributive algebras this solves the problem.

Problem : In strictly increasing algebras these junk routes can constantly be regenerated!

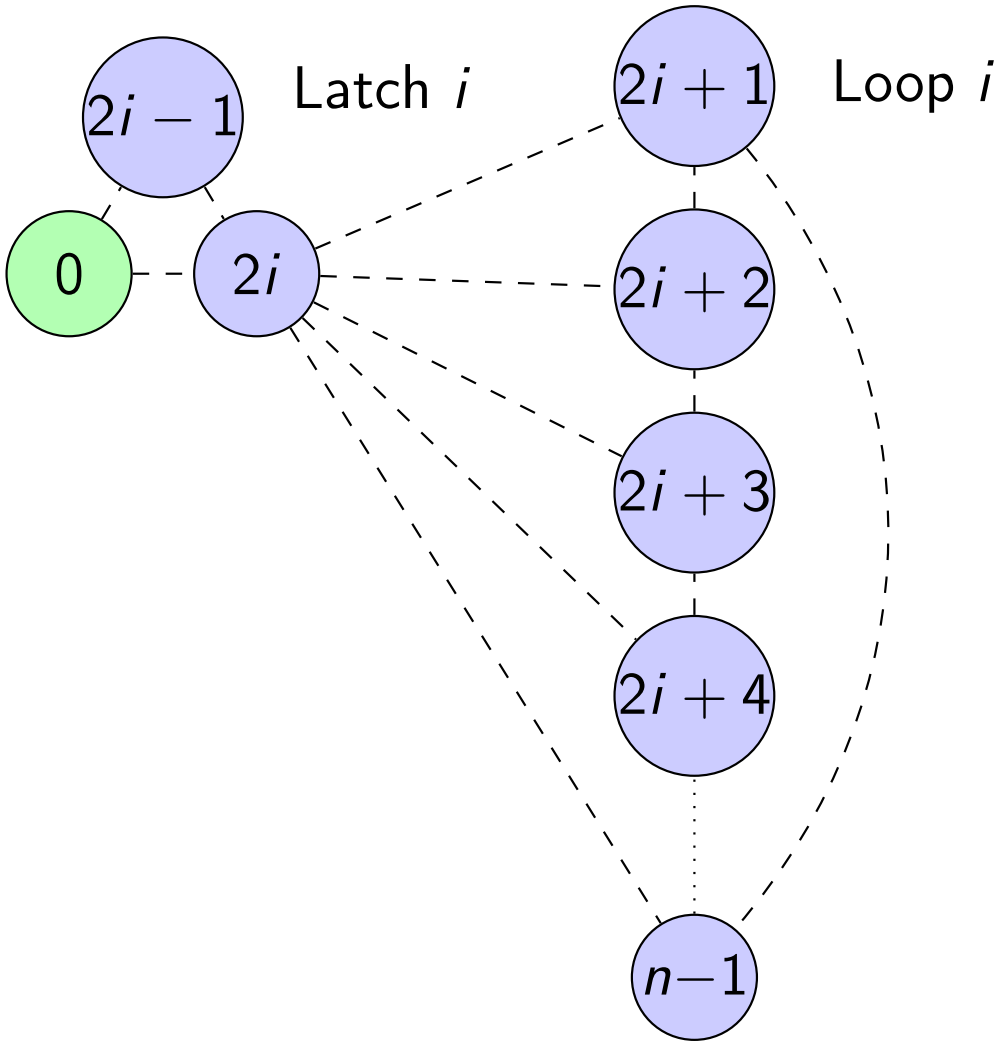
The Question

For a strictly increasing path-vector algebra, how many iterations of σ are required?

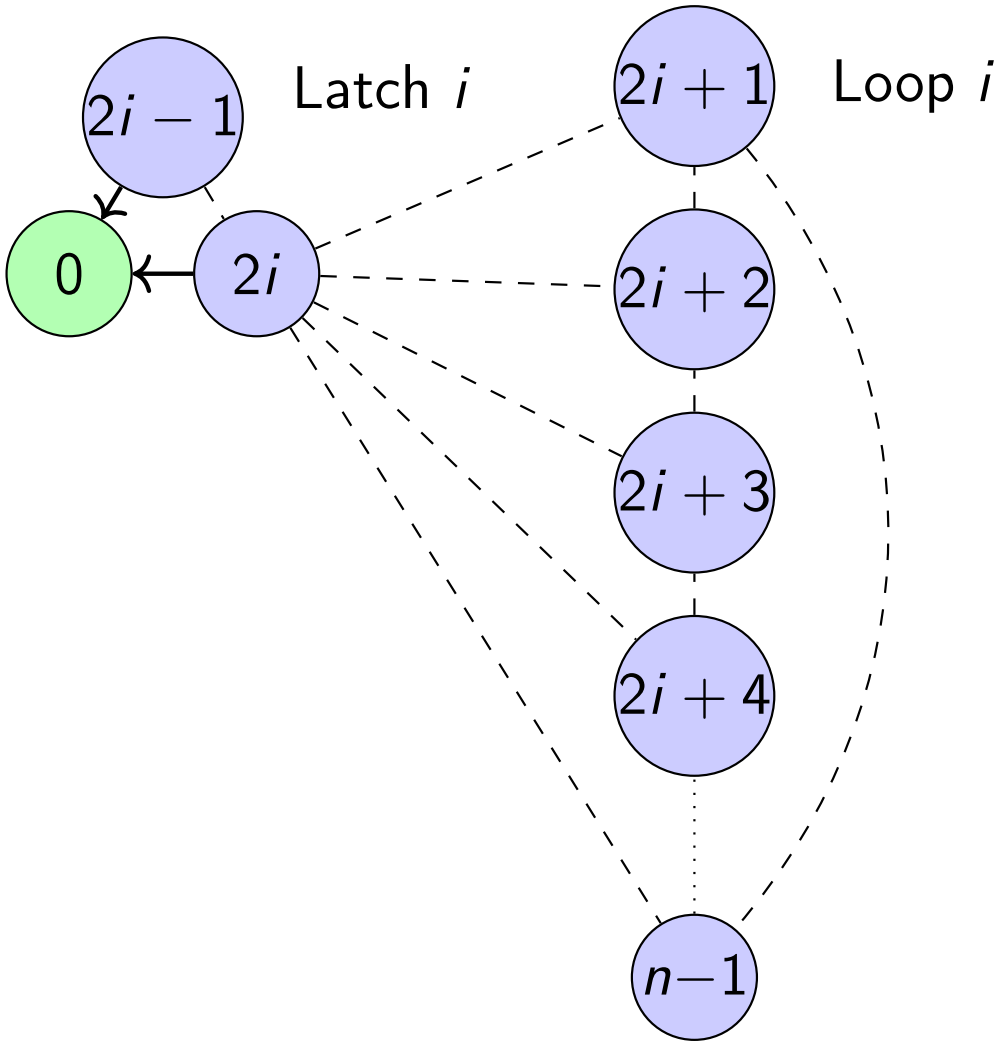
Our answer [ICNP 2018]

- For any $n > 4$ we construct a networks of size n that require $O(n^2)$ iterations.
- We show that no network can take more than $O(n^2)$ iterations.

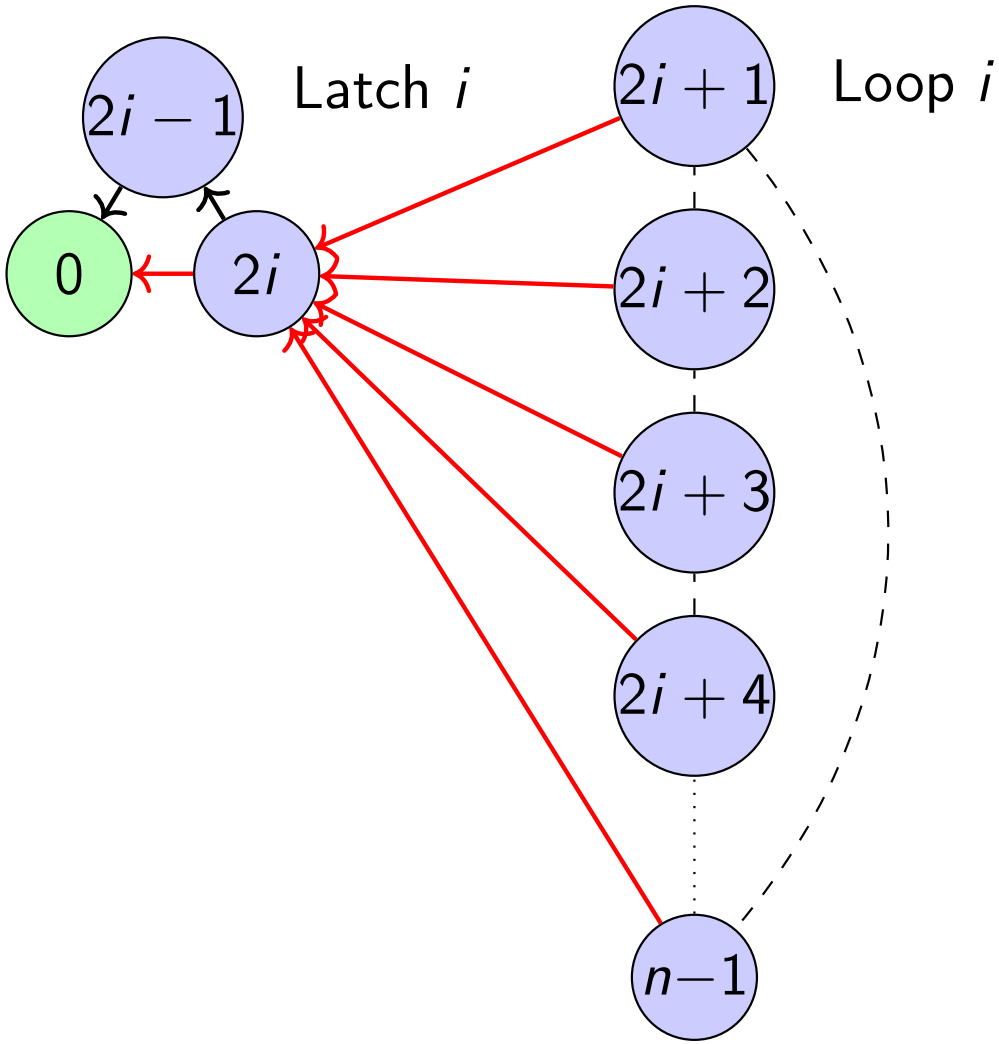
Gadgets – $O(n)$ propagation of junk routes



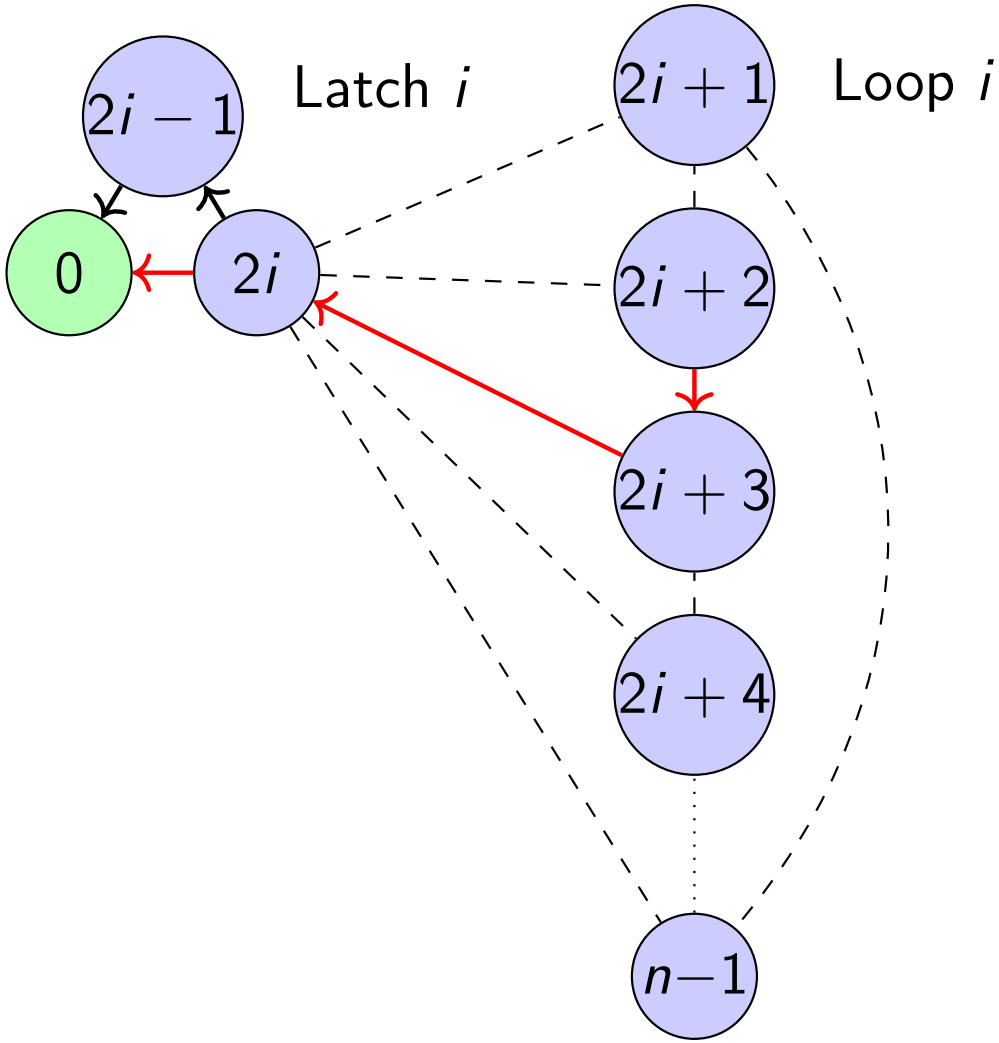
Gadgets – $O(n)$ propagation of junk routes



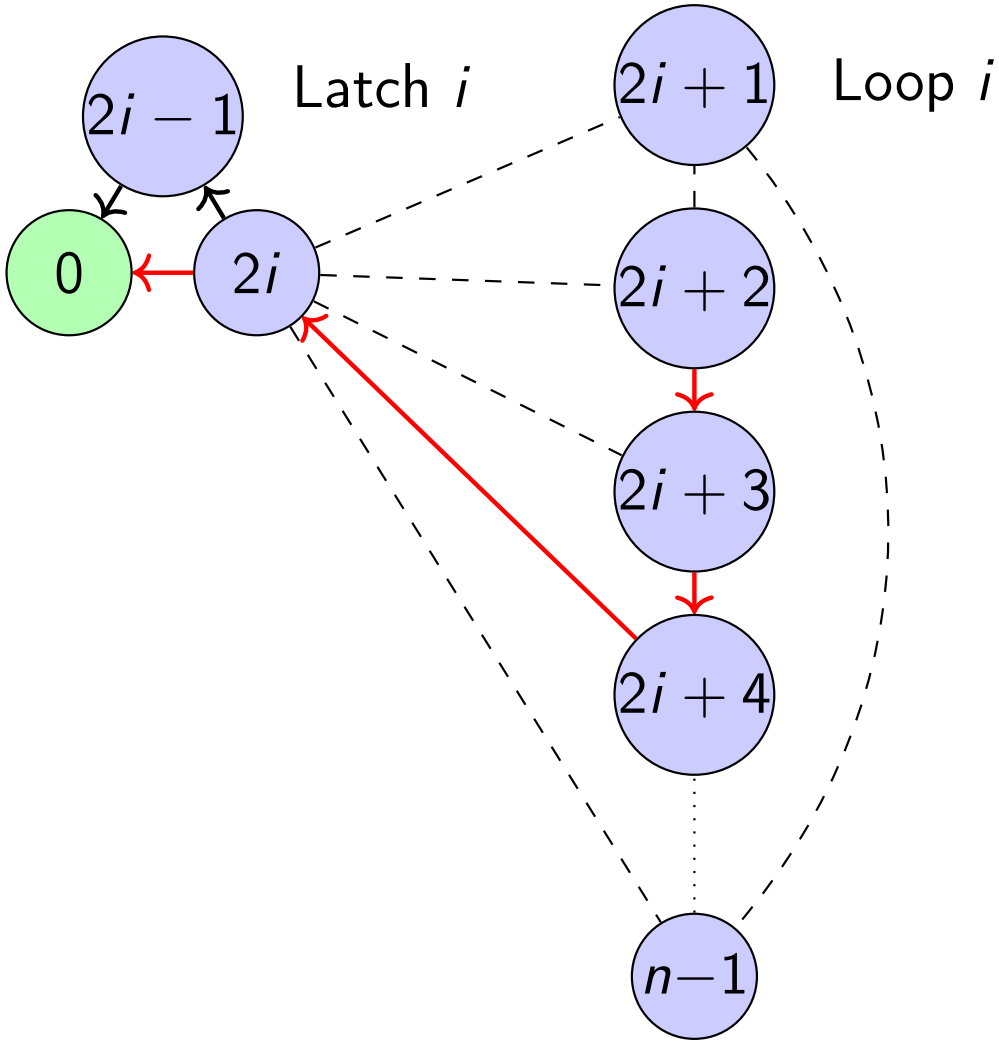
Gadgets – $O(n)$ propagation of junk routes



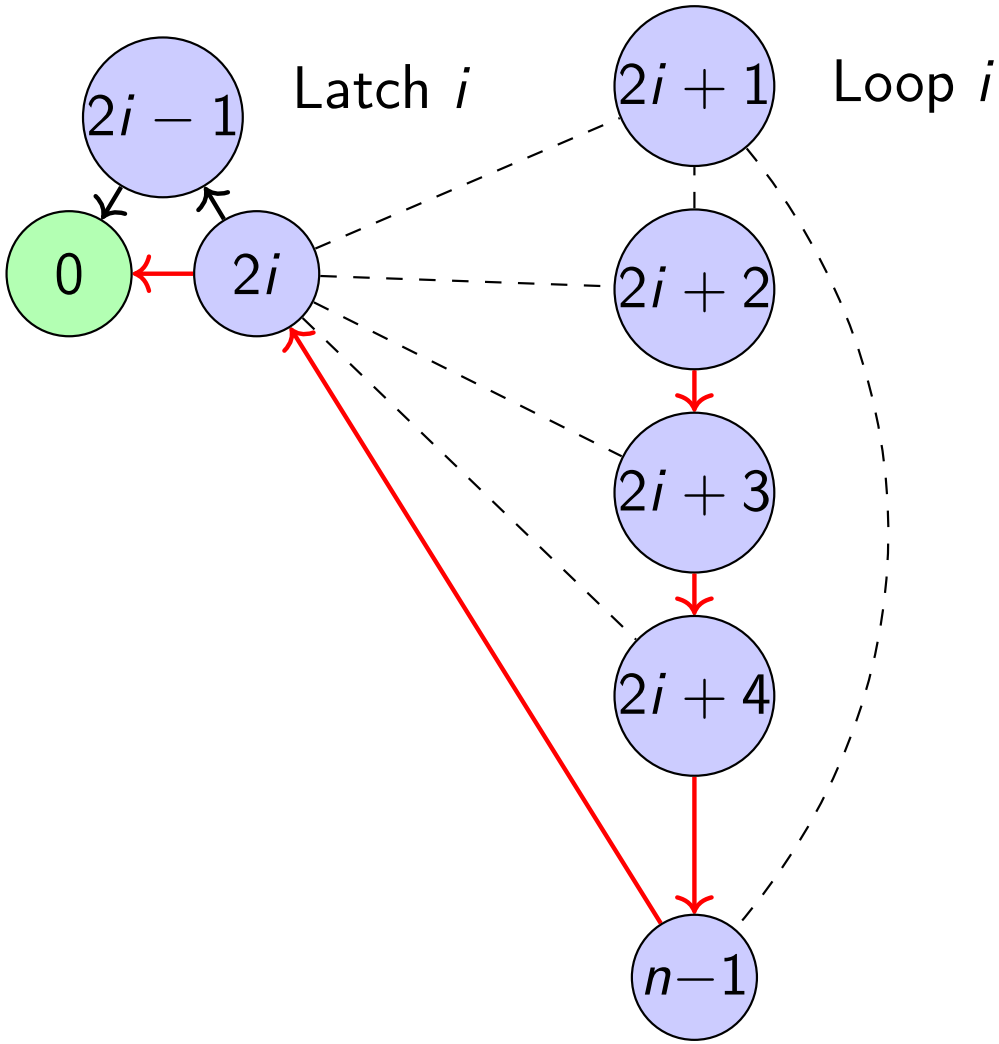
Gadgets – $O(n)$ propagation of junk routes



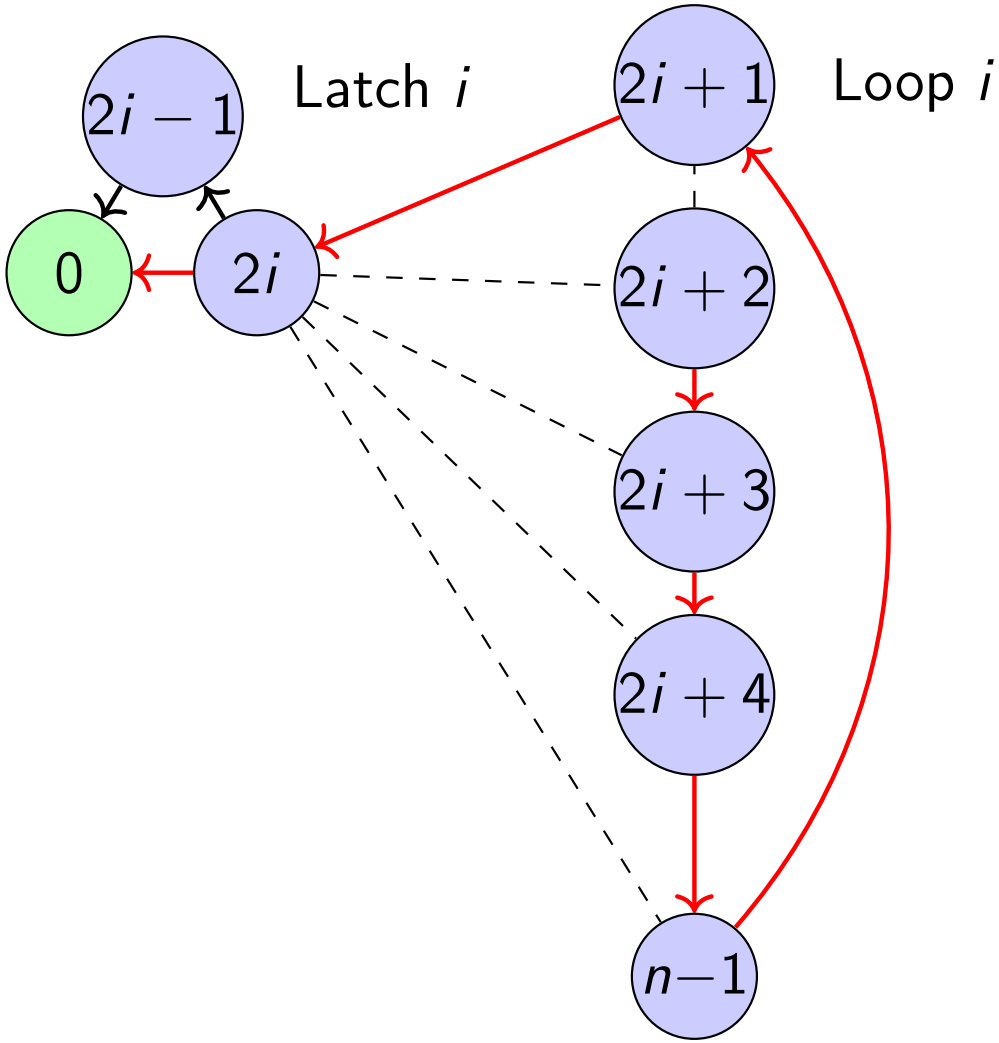
Gadgets – $O(n)$ propagation of junk routes



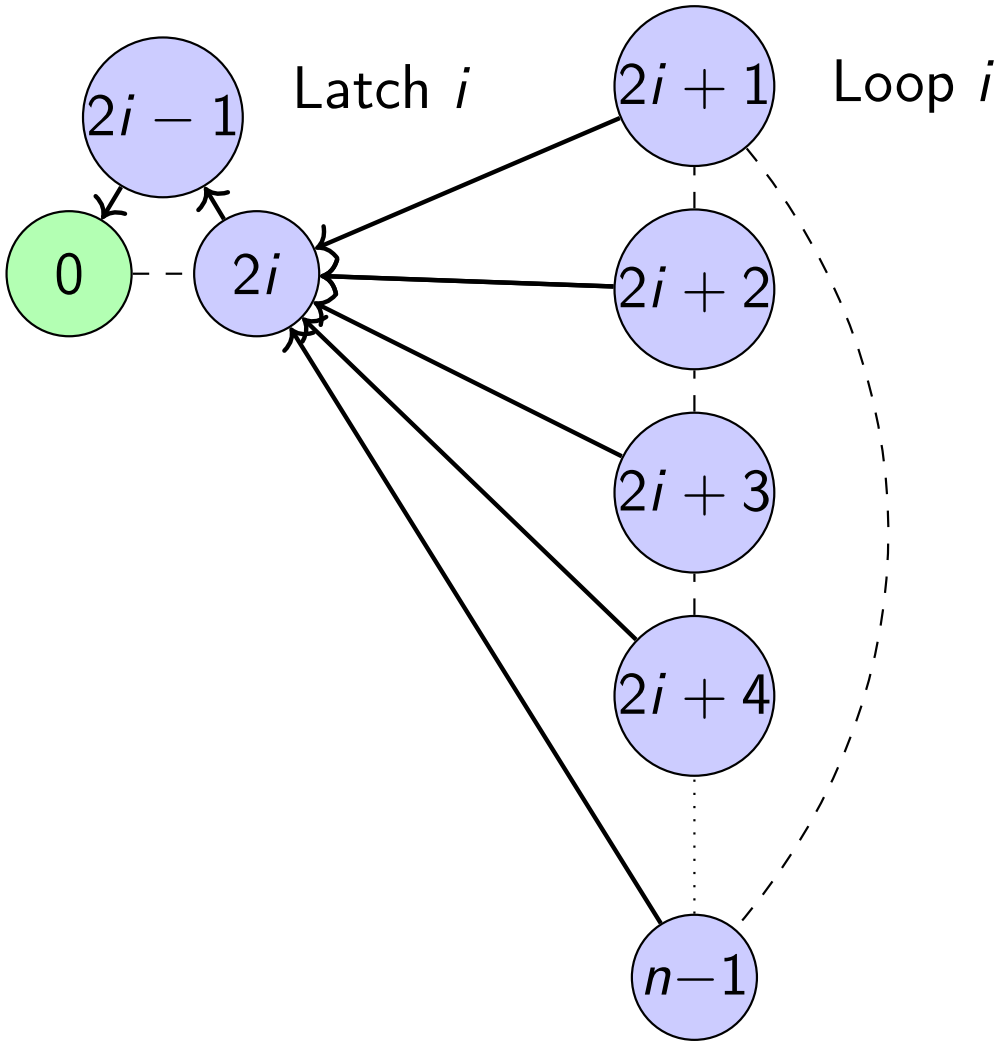
Gadgets – $O(n)$ propagation of junk routes



Gadgets – $O(n)$ propagation of junk routes

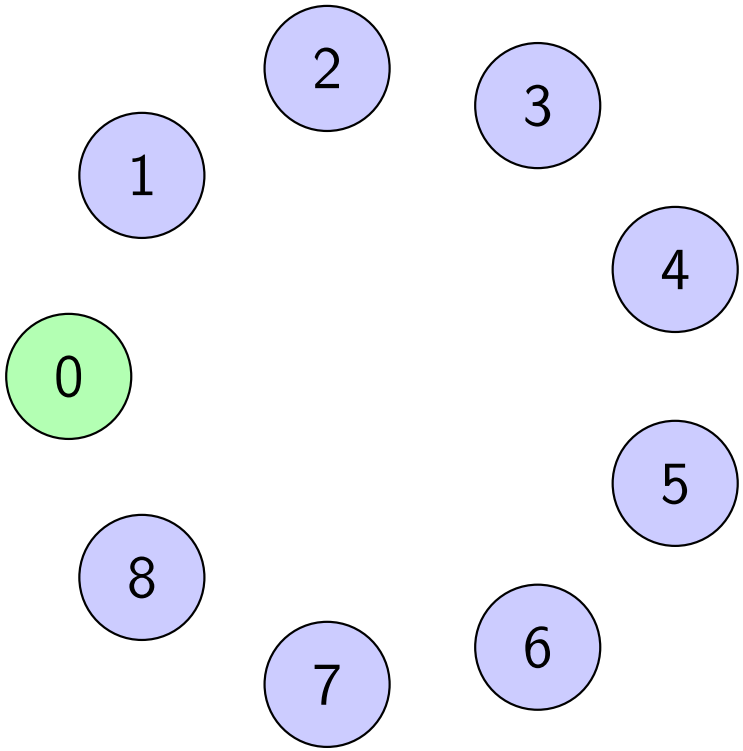


Gadgets – $O(n)$ propagation of junk routes



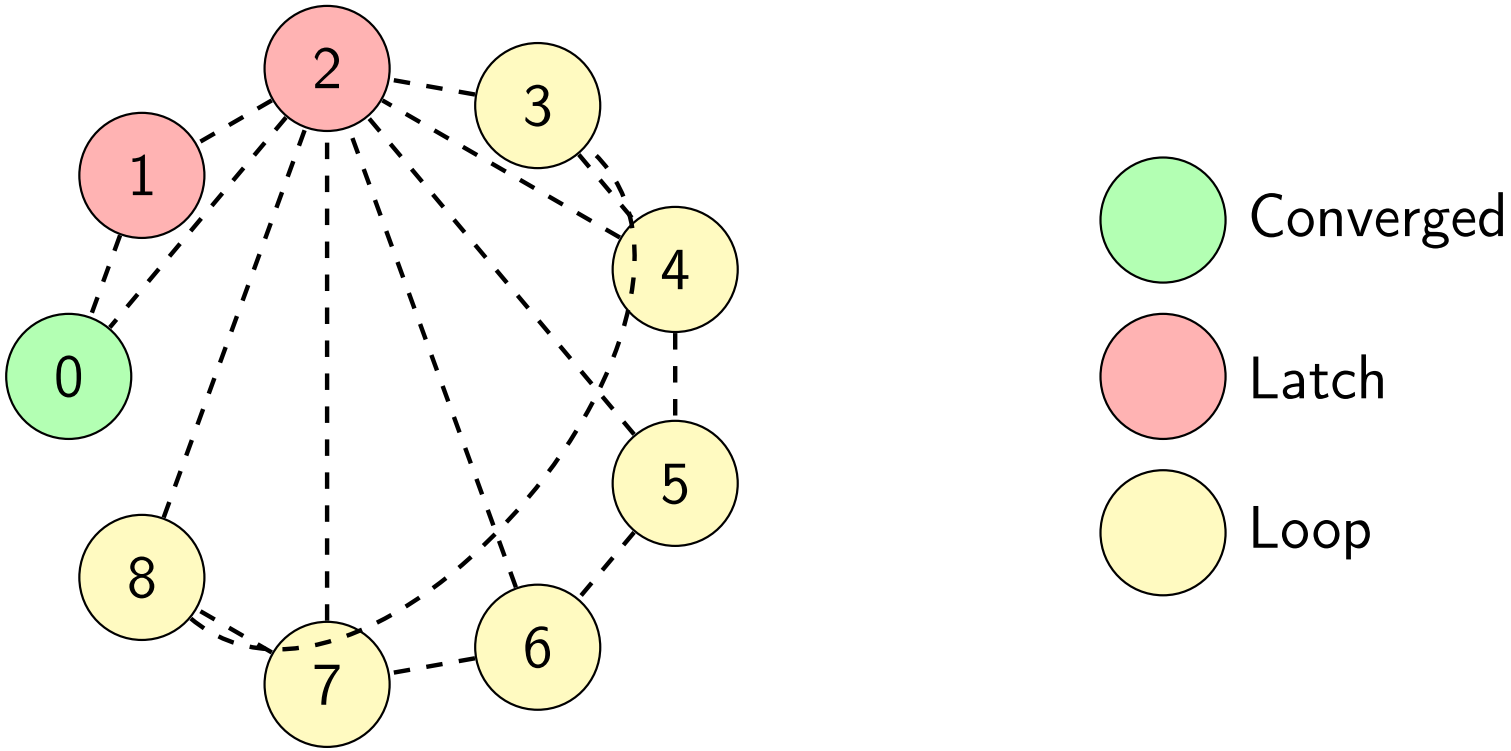
$O(n^2)$ operation

Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



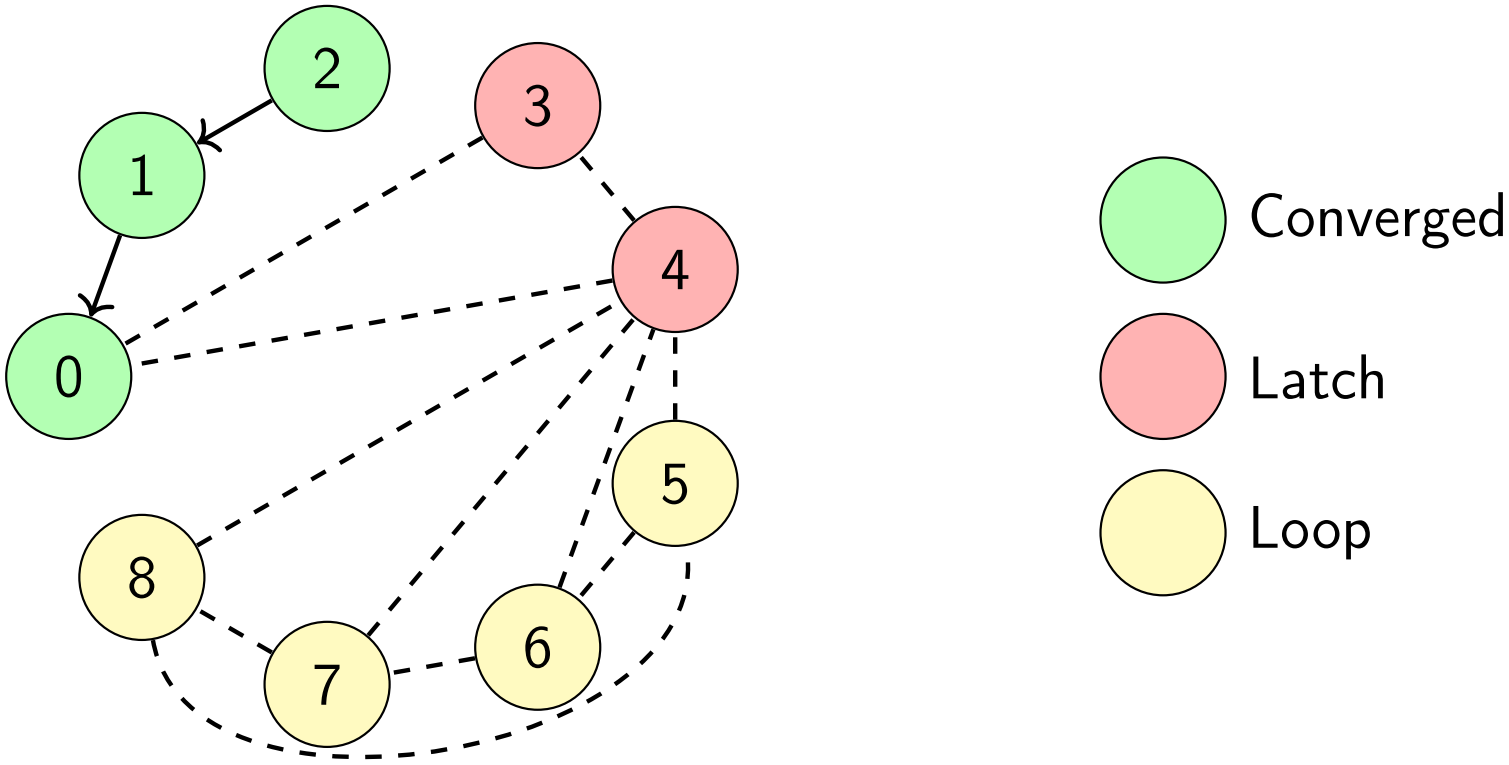
$O(n^2)$ operation

Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



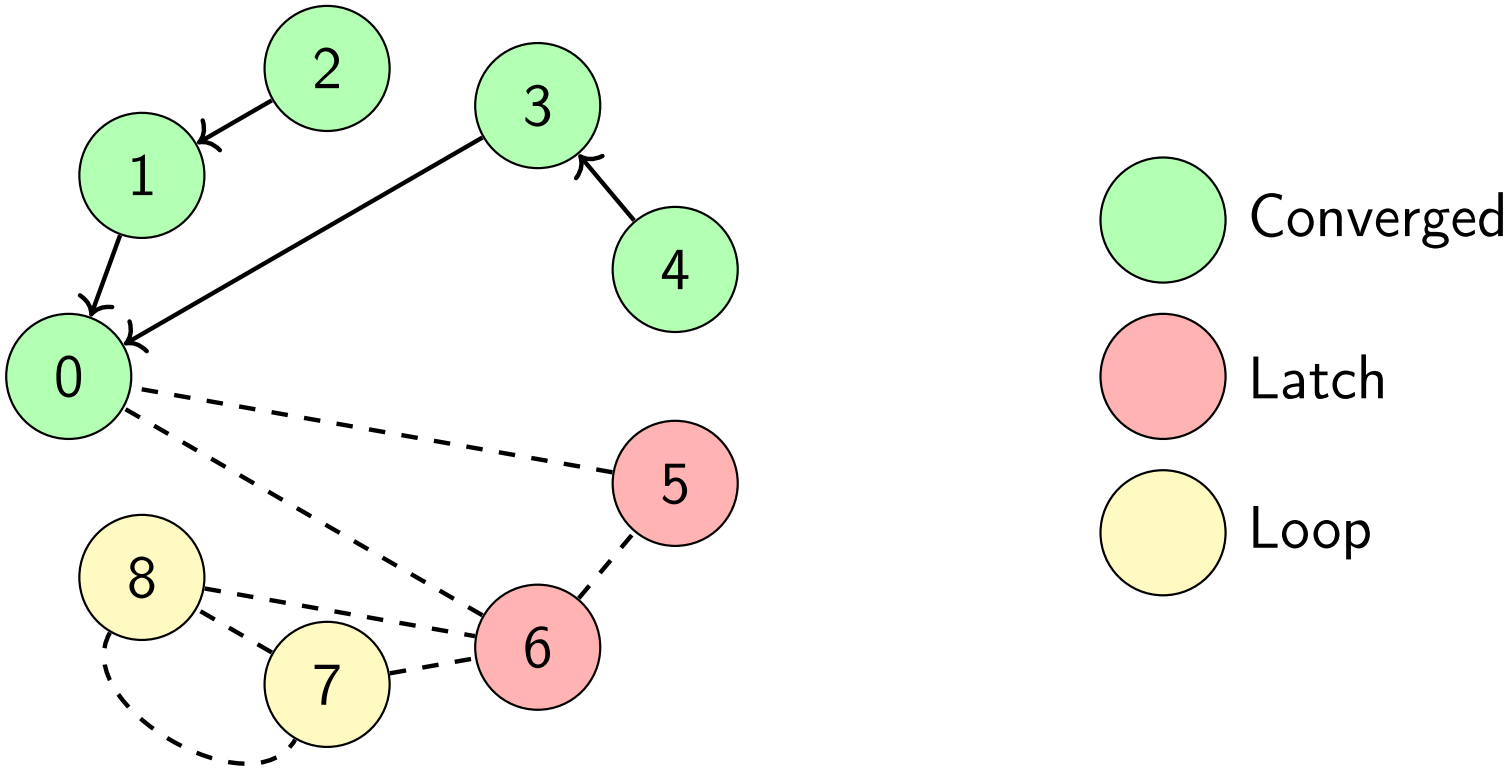
$O(n^2)$ operation

Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



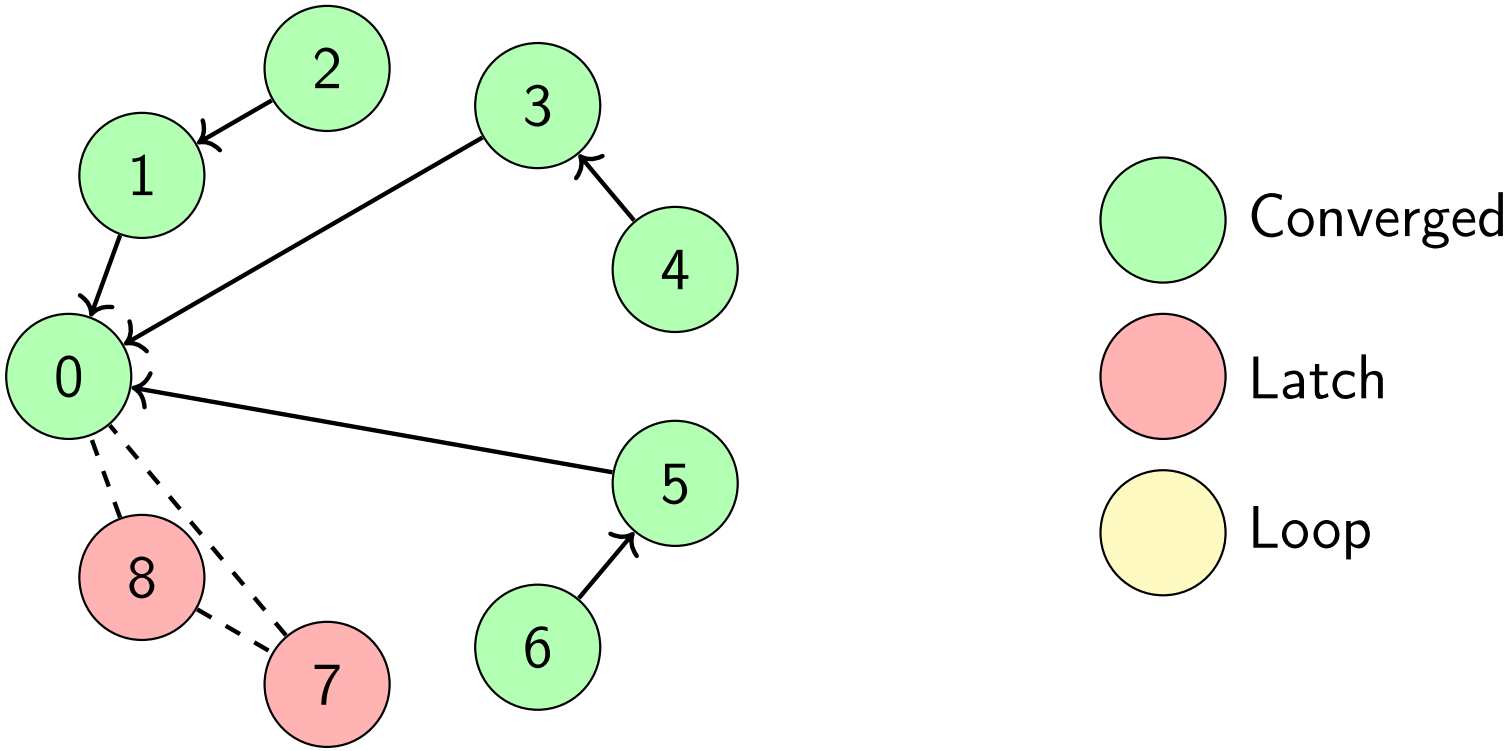
$O(n^2)$ operation

Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



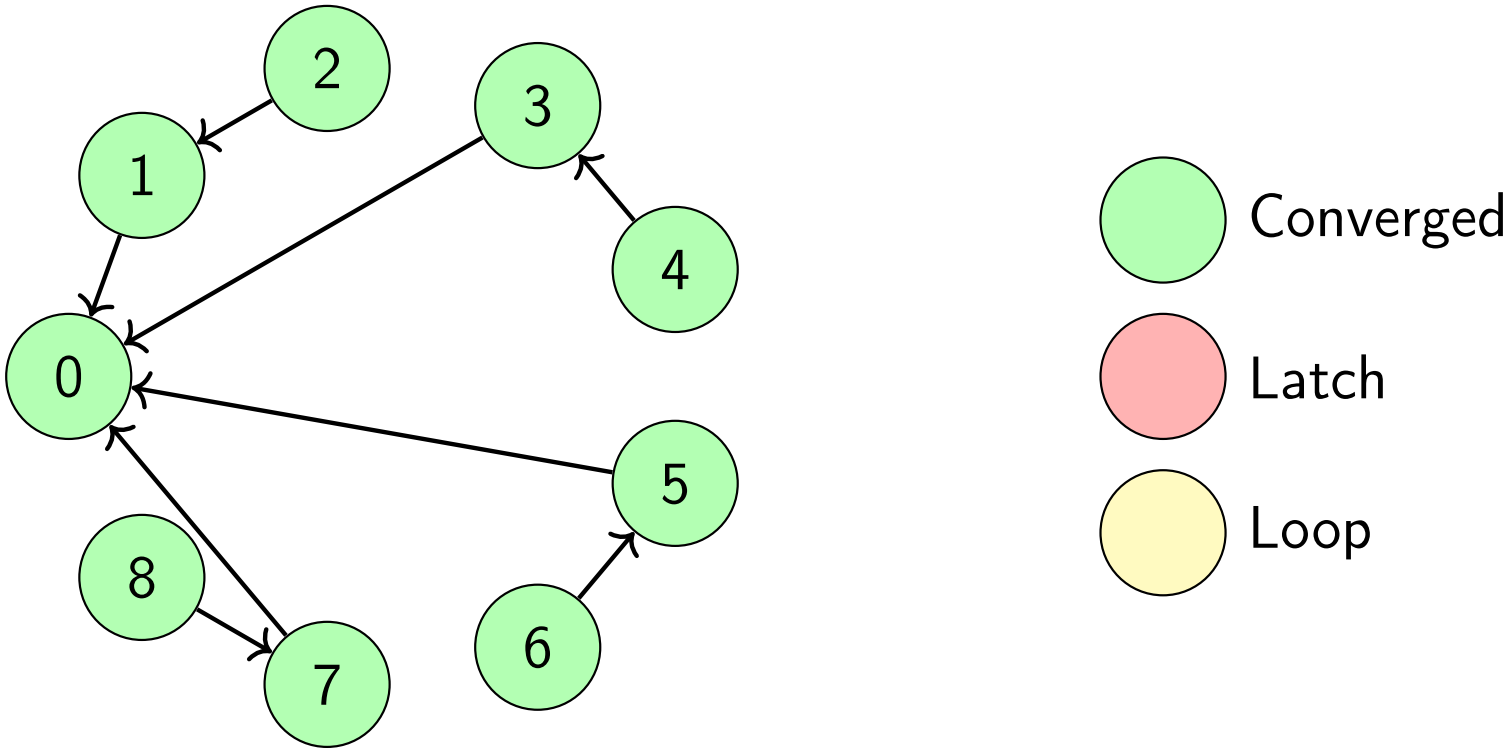
$O(n^2)$ operation

Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



$O(n^2)$ operation

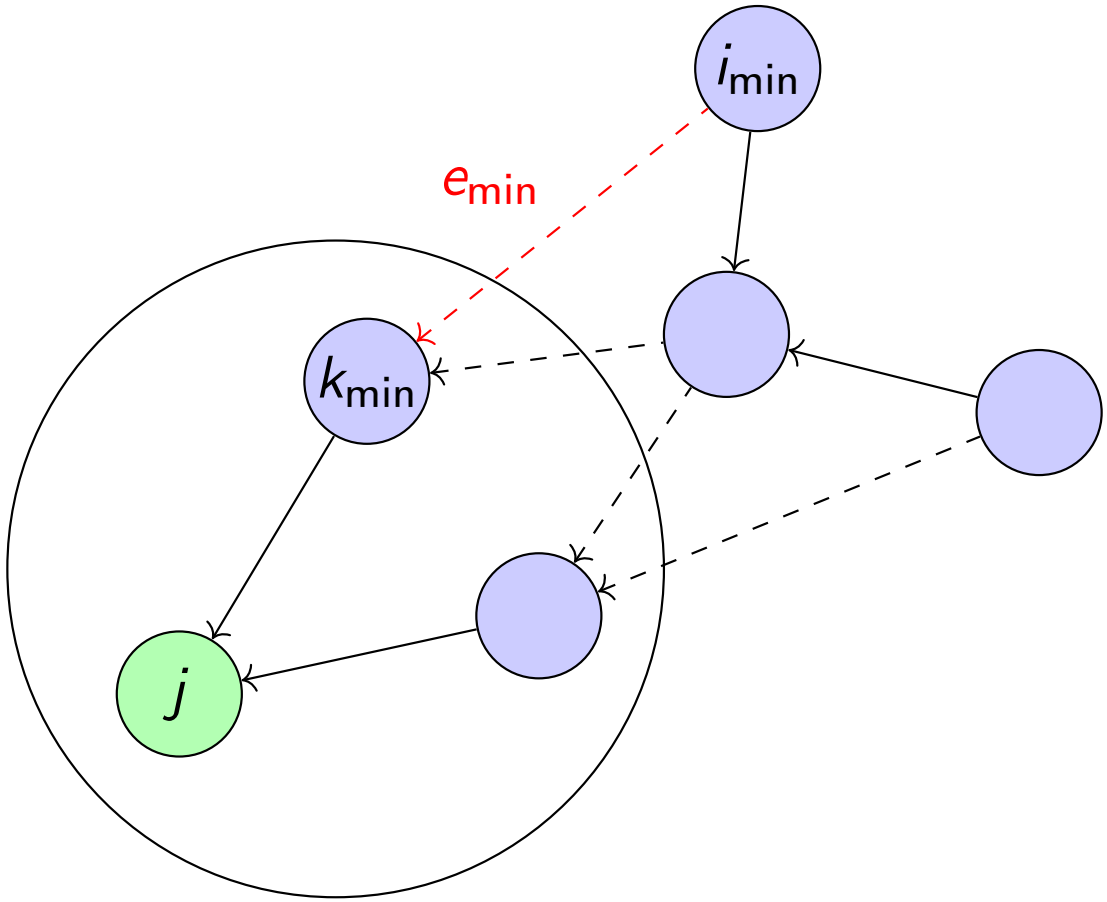
Quadratic convergence time achieved by nesting $O(n)$ gadgets with each gadget itself having $O(n)$ convergence time.



- The routing algebra and link weights that implement this scheme can be found in INCP 2018.
- For a network of n nodes the number of iterations required is $O(n^2)$.
- Is this truly the worst case?

Proof sketch

In fact guaranteed to converge in $< n^2$ iterations.



Quem ama piqui?

From iterative to asynchronous

Suppose we have an iterative algorithm $\mathbf{F}$ and an initial state $\mathbf{x}(0)$ where successive states are computed as

$$\mathbf{x}(t+1) = \mathbf{F}(\mathbf{x}(t))$$

until a fixed point ξ is reached at some time t' when $\mathbf{x}(t') = \xi = \mathbf{F}(\xi)$.

Assume that $\mathbf{x}(t)$ represents an n-dimensional vector in some state space. If we can express $\mathbf{F}$ as

$$\mathbf{F}(\mathbf{x}) = (\mathbf{F}_1(\mathbf{x}),\ \ldots,\ \mathbf{F}_n(\mathbf{x})),$$

then we can imagine that it may be possible to assign the computation of each $\mathbf{F}_i$ to a distinct processor in an asynchronous computation.

Note : in our case $\mathbf{F}$ is σ and each $\mathbf{F}_i$ is computed at a router i.

From iterative to asynchronous

Question

When can we use the $\mathbf{F}_i$ to correctly implement an *asynchronous* version of $\mathbf{F}$-iteration?

One answer

U&D

Üresin, A., Dubois, M.: Parallel asynchronous algorithms for discrete data. Journal of the ACM 37(3), 588–606 (Jul 1990)

Their results : if $\mathbf{F}$ is an Asynchronously Contracting Operator (ACO), then the any sensible asynchronous implementation of

$$\mathbf{F}(\mathbf{x}) = (\mathbf{F}_1(\mathbf{x}),\ \ldots,\ \mathbf{F}_n(\mathbf{x})),$$

will eventually converge to the fixed point of $\mathbf{F}$.

Applications of U&D

The good news

This allows us to avoid very difficult reasoning about asynchronous computations. Just reason about the **synchronous** function **F**.

The bad news

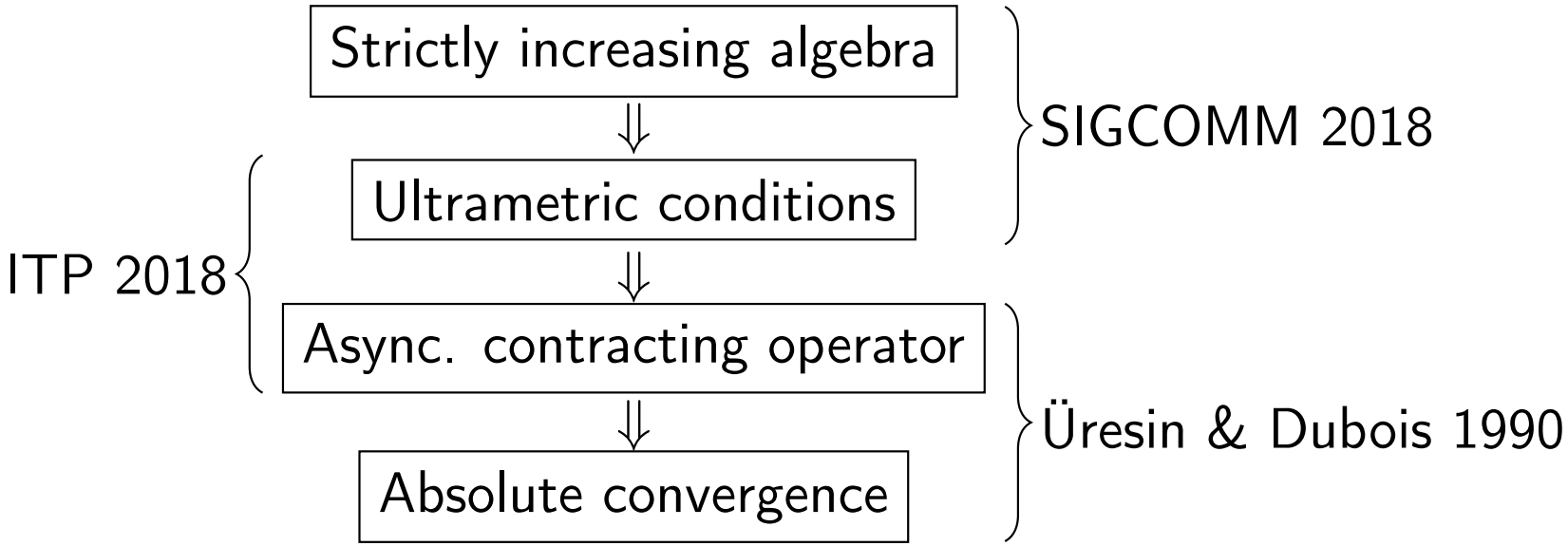
Proving that a given **F** is an ACO can be very difficult.

Alexander Gurney 2017

Gurney, A.J.T.: Asynchronous iterations in ultrametric spaces. Tech. rep. (2017), https://arxiv.org/abs/1701.07434

Results: A new set of sufficient conditions, based on ultra-metric theory, that imply ACO.

Putting it all together



This talk will not go into the details. If interested please read the papers!

Today's Tutorial

10:30-12:00 and **14:00-16:00** in **Bromélia**

Adventures in Algebraic Path Problems

- Part I : Classical Semiring-based path finding
- Part II : Drop distributivity. Show that Dijkstra's algorithm computes local optima (Sobrinho & Griffin 2010)

Thank you!

- Matthew Daggitt's Agda library
  - `https://github.com/MatthewDaggitt/agda-routing`
- Slides for this talk and today's tutorial
  - `https://github.com/Timothy-G-Griffin/SBRC_2019`